METHODS, SYSTEMS AND COMPUTER PROGRAM PRODUCTS FOR POLICY BASED DISTRIBUTION OF WORKLOAD TO SUBSETS OF POTENTIAL SERVERS

Related Applications

The present application is related to concurrently filed and commonly assigned United States Patent Application Serial No. _____, entitled "METHODS, SYSTEMS AND COMPUTER PROGRAM PRODUCTS FOR WORKLOAD DISTRIBUTION BASED ON END-TO-END QUALITY OF SERVICE" (Attorney Docket No. 5577-214), the disclosure of which is incorporated herein by reference as if set forth fully herein.

Field of the Invention

The present invention relates to network communications and more particularly to workload distribution within a cluster of data processing systems
5   communicating over a network.

Background of the Invention

Scalability and load balancing in network servers are issues which have received considerable attention in
10   light of the expansion of the Internet. For example, it may be desirable to have multiple servers servicing

customers. The workload of such servers may be balanced by providing a single network visible Internet Protocol (IP) address which is mapped to multiple servers.

Such a mapping process may be achieved by, for example, network address translation (NAT) facilities, dispatcher systems and Dynamic Name Server/WorkLoad Management (DNS/WLM) systems from International Business Machines Corporation (IBM), Armonk, New York. These various mechanisms for allowing multiple servers to share a single IP address are illustrated in **Figures** **1** through **3**.

Figure 1 illustrates a conventional network address translation system as described above. In the system of **Figure 1**, a client **10** communicates over a network **12** to a network address translation (NAT) system **14**. The network address translation system receives the communications from the client **10** and converts the communications from the addressing scheme of the network **12** to the addressing scheme of the network **12'** and sends the messages to the servers **16**. A server **16** may be selected from multiple servers **16** at connect time and may be on any host, one or more hops away. All inbound and outbound traffic flows through the NAT system **14**.

Figure 2 illustrates a conventional DNS/WLM system as described above. The server **16** is selected at name resolution time when the client **10** resolves the name for the destination server from the DNS/WLM system **17** which is connected to the servers **16** through the coupling facility (CF) **19**. The DNS/WLM system **17** of **Figure 2** relies on the client **10** adhering to a "zero time to live"

lifetime for IP addresses such that IP addresses are not cached by the client.

Figure 3 illustrates a conventional dispatcher system. As seen in **Figure** 3, the client **10** communicates over the network **12** with a dispatcher system **18** to establish a connection. The dispatcher routes inbound packets to the servers **16** and outbound packets are sent over network **12'** but may flow over any available path to the client **10**. The servers **16** are typically on a directly connected network to the dispatcher **18** and a server **16** is selected at connect time.

Such a dispatcher system is illustrated by the Interactive Network Dispatcher function of the IBM 2216 and AIX platforms. In these systems, the same IP address that the Network Dispatcher node **18** advertises to the routing network **12** is activated on server nodes **16** as loopback addresses. The node performing the distribution function connects to the endpoint stack via a single hop connection because normal routing protocols typically cannot be used to get a connection request from the endpoint to the distributing node if the endpoint uses the same IP address as the distributing node advertises. Network Dispatcher uses an application on the server to query a workload management function (such as WLM of System/390), and collects this information at intervals, *e.g.* 30 seconds or so. Applications running on the Network Dispatcher node **18** can also issue "null" queries to selected application server instances as a means of determining server instance health.

In addition to the above described systems, Cisco Systems offers a Multi-Node Load Balancing function on certain of its routers that performs the distribution function. Such operations appear similar to those of the

5   IBM 2216.

Finally, in addition to the system described above, AceDirector from Alteon provides a virtual IP address and performs network address translation to a real address of a selected server application. AceDirector appears to

10   observe connection request turnaround times and rejection as a mechanism for determining server load capabilities.

Summary of the Invention

Methods, systems and computer program products

15   according to embodiments of the present invention provide for distributing workload between a plurality of data processing systems in a cluster of data processing systems, wherein each of the plurality of data processing systems is executing an instance of an application which

20   communicates over a network such that a connection request to the application may be distributed to any one of the plurality of data processing systems. A subset of the plurality of data processing systems is defined which are to receive connection requests to the application

25   having at least one predefined characteristic. A request for a connection to the application is received over the network and it is determined if the request has a characteristic corresponding to the characteristic associated with the subset of the plurality of data

30   processing systems. The request is distributed to a data processing system in the subset of the plurality of data

processing systems if the request has a characteristic corresponding to the predefined characteristic.

In further embodiments of the present invention, the request is distributed to a data processing system of the plurality of data processing systems other than one in the subset of data processing systems if the request does not have a characteristic corresponding to the predefined characteristic. The characteristic may be a client identification associated with the request which may be a source address of a request.

In still further embodiments of the present invention, the availability of data processing systems in the subset of the plurality of data processing systems is determined. It is also determined if the availability of at least one of the data processing systems in the subset of the plurality of data processing systems meets an availability criteria. If so, the request is distributed to a data processing system in the subset of the plurality of data processing systems which meets the availability criteria.

Furthermore, the request may be distributed to a data processing system of the plurality of data processing systems other than a data processing system in the subset of data processing systems if the request has a characteristic corresponding to the predefined characteristic and none of the subset of the plurality of data processing systems have an availability which meets the availability criteria. Alternatively, the request may be rejected if the request has a characteristic corresponding to the predefined characteristic of the subset of the plurality of data processing systems and

none of the subset of the plurality of data processing systems have an availability which meets the availability criteria.

In still further embodiments of the present invention, an availability of data processing systems other than data processing systems in the subset of data processing systems is determined. The request is distributed to a data processing system of the plurality of data processing having the best availability if the request has a characteristic corresponding to the at least one predefined characteristic of the subset of the plurality of data processing systems and none of the subset of the plurality of data processing systems have an availability which meets the availability criteria.

In additional embodiments of the present invention, a subset of the plurality of data processing systems may be defined which includes data processing systems having common operational characteristics. The data processing systems may be communication protocol stacks in an OS/390 Sysplex to which the application is bound. In such embodiments, the subset of the plurality of data processing systems may be a subset of the communication protocol stacks bound to the application. Furthermore, the distribution of the requests may be provided by a routing communication protocol stack.

As will further be appreciated by those of skill in the art, the present invention may be embodied as methods, apparatus/systems and/or computer program products.

Brief Description of the Drawings

Figure 1 is block diagram of a conventional network address translation system;

Figure 2 is block diagram of a conventional DNS/WLM system;

Figure 3 is block diagram of a conventional dispatcher system;

Figure 4 is a block diagram illustrating embodiments of the present invention;

Figure 5 is a flowchart illustrating operations according to embodiments of the present invention;

Figure 6 is a flowchart illustrating operations according to embodiments of the present invention;

Figure 7 is block diagram of a cluster of data processing systems incorporating embodiments of the present invention; and

Figure 8 is a flowchart illustrating operations of a routing protocol stack according to embodiments of the present invention.

Detailed Description of the Invention

The present invention now will be described more fully hereinafter with reference to the accompanying drawings, in which preferred embodiments of the invention are shown. This invention may, however, be embodied in many different forms and should not be construed as limited to the embodiments set forth herein; rather, these embodiments are provided so that this disclosure will be thorough and complete, and will fully convey the scope of the invention to those skilled in the art. Like numbers refer to like elements throughout.

As will be appreciated by those of skill in the art, the present invention can take the form of an entirely hardware embodiment, an entirely software (including firmware, resident software, micro-code, *etc.*)

5   embodiment, or an embodiment containing both software and hardware aspects. Furthermore, the present invention can take the form of a computer program product on a computer-usable or computer-readable storage medium having computer-usable or computer-readable program code

10   embodied in the medium for use by or in connection with an instruction execution system. In the context of this document, a computer-usable or computer-readable medium can be any structure that can contain, store, communicate, propagate, or transport the program for use

15   by or in connection with the instruction execution system, apparatus, or device.

The computer-usable or computer-readable medium can be, for example, but is not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or

20   semiconductor system, apparatus, device, or propagation medium. More specific examples (a nonexhaustive list) of the computer-readable medium would include the following: an electrical connection having one or more wires, a removable computer diskette, a random access memory

25   (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), an optical fiber, and a portable compact disc read-only memory (CD-ROM). Note that the computer-usable or computer-readable medium could even be paper or another suitable medium

30   upon which the program is printed, as the program can be electronically captured, via, for instance, optical

scanning of the paper or other medium, then compiled, interpreted, or otherwise processed in a suitable manner if necessary, and then stored in a computer memory.

The present invention can be embodied as systems, methods, or computer program products which allow for workload distribution between a plurality of communication protocol stacks in a cluster of data processing systems. While embodiments of the present invention may be incorporated into any routing based workload distribution system, in particular embodiments of the present invention, workload is distributed by routing protocol stacks which associate a Virtual IP Address (VIPA) and port with other communication protocol stacks in the cluster. The routing communication protocol stacks route communications to the VIPA and port to the appropriate communication protocol stack. VIPAs capable of being shared by a number of communication protocol stacks are referred to herein as "dynamic routable VIPAs" (DVIPA). Workload distribution according to embodiments of the present invention may be accomplished in such routing communication protocol stacks during the selection process when a connection using the DVIPA is established.

Particular preferred systems in which embodiments of the present invention may be utilized are described in commonly assigned United States Patent Application Serial No. 09/640,412, entitled, "METHODS, SYSTEMS AND COMPUTER PROGRAM PRODUCTS FOR NON-DISRUPTIVELY TRANSFERRING A VIRTUAL INTERNET PROTOCOL ADDRESS BETWEEN COMMUNICATION PROTOCOL STACKS" (Attorney Docket No. 5577-207), United States Patent Application Serial No. 09/640,409, entitled

"METHODS, SYSTEMS AND COMPUTER PROGRAM PRODUCTS FOR
CLUSTER WORKLOAD DISTRIBUTION" (Attorney Docket No. 5577-
205), United States Patent Application Serial No.
09/640,438, entitled "METHODS, SYSTEMS AND COMPUTER

5   PROGRAM PRODUCTS FOR FAILURE RECOVERY FOR ROUTED VIRTUAL
INTERNET PROTOCOL ADDRESSES" (Attorney Docket No. 5577-
206) all filed August 17, 2000, and United States Patent
Application Serial No. 09/401,419 entitled "METHODS,
SYSTEMS AND COMPUTER PROGRAM PRODUCTS FOR AUTOMATED

10   MOVEMENT OF IP ADDRESSES WITHIN A CLUSTER" filed
September 22, 1999, the disclosures of which are each
incorporated herein by reference as if set forth fully
herein.

        A system incorporating embodiments of the present

15   invention is illustrated in **Figure 4**. As seen in **Figure
4**, a workload distributer **50** communicates with a network
so as to receive connection requests for connections to
applications executing on data processing systems (target
servers) in a cluster of data processing systems. The

20   workload distributer **50** distributes the connection
requests to data processing systems which are executing
an application for which the connection is requested,
such as the target servers **56, 56', 56'', 58** and **60** which
are each executing an instance of a common application.

25   Other data processing systems may be included in the
cluster which are executing other applications, however,
for clarity of illustration, these data processing
systems are not shown in **Figure 4**. Furthermore, the
target servers may be executing instances of other

30   applications as well as different servers may be
executing different applications. Workload distribution

may also be carried out according to the teachings of the present invention for these other applications, however, for purposes of clarity the examples of the present invention are described with reference to one

5   application.

As is further illustrated in **Figure 4**, the workload distributer may distribute connection requests to the application executing on the target servers **56, 56', 56'', 58** and **60** utilizing the workload and quality of

10   service information **54** and utilizing workload distribution policies **52**. The workload distribution policies **52** may serve to establish subsets of available servers for a connection request to an application which are associated with one or more characteristics of the

15   connection request. As used herein, the term "available servers" refers to those servers for which the workload distributer **50** distributes connection requests which are capable of handling a particular connection request. Furthermore, the term "server" is used herein generally

20   to refer to an entity capable of executing an application for which connections may be requested and which may be a target for distribution of connection requests by the workload distributer **50**. Thus, examples of "servers" may include a data processing system or operating system

25   executing on a data processing system if multiple operating system images may execute on a single data processing system. For example, in a data processing system having multiple communication protocol stacks, if each communication protocol stack may receive

30   connections, then each such communication protocol stack

may be treated as a distinct "server" for purposes of workload distribution.

Thus, for example, a policy may establish a subset 62 including target servers 56, 56' and 56'' of the total potential target servers 56, 56', 56'', 58 and 60 for an application. The subset 62 may be associated with one or more connection characteristics, such as the identity of the client requesting the connection, such that connection requests having the one or more characteristics are preferentially distributed by the workload distributer 50 to the target servers 56, 56' and 56'' in the subset 62. Any characteristic which may be determined from the connection request and which may provide the basis for distinguishing between requests based on the characteristic may be utilized in determining if a connection request should be preferentially routed to a subset of the data processing systems. For example, application level information, such as information available above the TCP/IP layer (e.g. "layer 7" information), source Internet Protocol address, data payload type, client identifiers, security protocols or other such characteristics of the connection request may be utilized.

Furthermore, in certain embodiments of the present invention, the target servers 56, 56' and 56'' may be selected from the total potential target servers 56, 56', 56'', 58 and 60 for inclusion in the subset 62 based on common operational characteristics or based on having operational characteristics which fall within a common range. Examples of operational characteristics which may

be used to group data processing systems into subsets may include processing capabilities such as content location, processing speed, available memory or other performance criteria, communication capabilities, such as total available bandwidth, connection speed, connection type or the like, security issues or even ownership issues such as ownership and/or control of the data processing systems. In fact, subsets could be dynamically established based on real-time capabilities of particular servers (*e.g.*, servers with excess capacity may form a group to serve premium users *etc.*)

By allowing policies to preferentially direct or even to restrict the potential target servers to a subset of the total, differentiation between individual requests may be provided. Thus, for example, different service levels may be provided by evaluating requests and providing requests to subsets of available servers based on the service level indicated by the characteristics of a request.

In one example, if the servers are application service providers, then different levels of performance could be sold to different customers. These different performance levels may be associated with different subsets of application servers. Requests for connections to an application server could be evaluated to determine the performance level that a customer corresponding to the requesting client paid for and the connection assigned to the subset of application servers associated with that performance level. Similarly, in an e-commerce system, types of transactions could be segregated such that connections associated with purchases were handled

by higher performance machines than those associated with browsing of an on-line catalog.

Furthermore, different subsets of servers may be utilized based on the type of data associated with the connection. For example, if a connection request was for a connection to provide streaming audio or video a different subset of servers may be used than if the connection is for the download of a data file.

Additionally, when different types of devices request connections, the type of device requesting the connection could be used to select a subset of servers. For example, a subset of processors with low processing performance could be assigned to wireless devices which have lower bandwidth communication capabilities. Connections requested by desktop systems with higher bandwidth communication capabilities could result in the selection of a subset of servers with higher processing performance. As is seen from the above discussion, embodiments of the present invention may be used in a variety of situations and/or environments with subsets of servers selected in a different manner based on the particular situation and/or environment.

Operations of the workload distributer **50** according to embodiments of the present invention will now be described with reference to **Figure 5**. The workload distributer **50** defines a subset of available servers for connections to an application (block **80**). Such a definition may be based on the policies **52** of the workload distributer **50**. The subsets of available servers are associated with one or more characteristics

of connection requests (block **82**). Such an association may also be provided by the policies **52**. Furthermore, the characteristics of connection requests may be as described above.

5   When a connection request is received, the connection request is evaluated to determine the characteristics of the connection request (block **84**). Such an evaluation may take different forms based on the characteristics which are to be evaluated. For example,

10   the evaluation may be evaluating the source IP address of the connection request or may be evaluating the data payload to determine the type of data for which the connection is requested (*e.g.*, streaming video, streaming audio, file download, *etc.*).

15   Based on the determined characteristics of the connection request (block **84**), the workload distributer **50** selects a server from the subset **62** of servers corresponding to the characteristics of the connection request (block **86**). The connection request is

20   distributed by the workload distributer **50** to the selected server (block **88**).

Operations according to embodiments of the present invention which utilize policies **52** and workload and QoS information **54** will now be described with reference to

25   **Figure 6**. As seen in **Figure 6**, a request for a connection to an application executing on data processing systems in a cluster of data processing systems is received (block **100**). For example, the request may be received by the workload distributer **50**. The connection

30   request is evaluated to determine if the request has a

characteristic which is associated with a subset of the
data processing systems executing the application (block
101). If the connection request has a characteristic
which is associated with a subset of the data processing

5   systems executing the application (block **101**), then that
subset of the data processing systems is set as the
target servers (block **103**). If the connection request
does not have a characteristic which is associated with a
subset of the data processing systems executing the

10   application (block **101**), then all available servers may
be set as the target servers (block **105**).

        Workload information, such as an indication of the
processing capacity utilization of the target servers in
the cluster of data processing servers, is obtained

15   (block **102**) as well as QoS information for the target
servers (block **104**). The QoS information may be obtained
on a server basis or for connections having constraints
in common with the requested connection (*e.g.* a common
QoS level). The processing capacity utilization

20   information and the QoS information may be utilized
(block **106**) to provide a workload metric for use in
selecting a target server for the connection. The target
server having the best or a sufficiently satisfactory
workload metric is selected (block **108**) and the

25   connection request is distributed to the selected target
server (block **110**). Alternatively, if no target servers
in the subset are available, the connection request could
be distributed to target servers outside the subset of
target servers.

The QoS information utilized in the workload distribution preferably includes QoS information such as packet loss information, timeout information and/or connection usage information. Other suitable QoS

5   measurements may also be utilized. The QoS measurements may be ratios, such as a packet loss ratio, a timeout ratio, or a connection utilization ratio, which may be used directly or may be weighted. Workload information may, for example, be provided as a workload weight (W)

10   which expresses the utilization of capacity for a target server as a range of values. For example, the IBM WorkLoad Manager (WLM) provides an integer value from 0 to 64 with 64 being the highest availability capacity. Such workload values may represent the capacity of the

15   server, the application, other critical resources of the server or combinations thereof. Furthermore, while the combination of QoS information and processor capacity utilization information is described above as being performed as a separate operation, as will be appreciated

20   by those of skill in the art, such operations need not produce an intermediate result but may be combined in a single operation.

Particular embodiments of the present invention will now be described with reference to **Figures** 7 and **8** which

25   illustrate embodiments of the present invention in a Sysplex cluster. A cluster of data processing systems is illustrated in **Figure 7** as cluster nodes in the Sysplex **10**. As seen in **Figure 7**, several data processing systems **20**, **24**, **28**, **32** and **36** are interconnected in a

30   Sysplex **10**. The data processing systems **20**, **24**, **28**, **32** and **36** illustrated in **Figure 7** may be operating system

images, such as MVS images, executing on one or more
computer systems. While the present invention will be
described primarily with respect to the MVS operating
system executing in a System/390 environment, the data

5   processing systems **20, 24, 28, 32** and **36** may be mainframe
computers, mid-range computers, servers or other systems
capable of supporting workload distribution based on
workload and end-to-end connection QoS information as
described herein.

10   As is further illustrated in **Figure** 7, the data
processing systems **20, 24, 28, 32** and **36** have associated
with them communication protocol stacks **22, 26, 30, 34**
and **38**, which may be TCP/IP stacks. The communication
protocol stacks **22, 26, 30, 34** and **38** incorporate a VIPA

15   distribution function **23** as described in the above
reference United States Patent Applications.
Furthermore, the communication protocol stacks **22** and **38**
function as routing communication protocol stacks and,
therefore, also incorporate a workload distribution

20   function **25** which may function as a workload distributer
and which may work in cooperation with the VIPA
distribution function **23** so as to distribute DVIPA
connections among target communication protocol stacks
based on resource availability and end-to-end QoS

25   information. The communication protocol stacks **26, 30**
and **34** function as server communication protocol stacks
and, therefore, include a policy agent **27** which provides
QoS information to the workload distribution function **25**.
If a communication protocol stack functions as both a

30   server stack and a routing stack, then the workload

distribution function **25** may perform the functions of the
policy agent **27** or a policy agent **27** may also be provided
on these stacks.

While each of the communication protocol stacks **22,**
5   **26, 30, 34** and **38** illustrated in **Figure** 7 incorporate the
VIPA distribution function **23** and either a workload
distribution function **25** or a policy agent **27**, not all
communication protocol stacks in a Sysplex need
incorporate these functions. Thus, the present invention
10   may be carried out on any system where two or more
communication protocol stacks in a cluster of data
processing systems support workload distribution and
dynamic routable VIPAs. If a communication protocol
stack does not support workload distribution and dynamic
15   routable VIPAs, then the workload distribution and
dynamic routable VIPA messages according to the present
invention could be ignored by the communication protocol
stack. Thus, the present invention may provide backward
compatibility with existing communication protocol
20   stacks.

As is further seen in **Figure** 7, the communication
protocol stacks **22, 26, 30, 34** and **38** may communicate
with each other through a coupling facility **40** of the
Sysplex **10**, for example, utilizing XCF messaging.
25   Furthermore, the communication protocol stacks **22** and **38**
may communicate with an external network **44** such as the
Internet, an intranet, a Local Area Network (LAN) or Wide
Area Network (WAN) utilizing the Enterprise System
Connectivity (ESCON) **42**. Thus, a client **46** may utilize
30   network **44** to communicate with an application executing

on an MVS image in Sysplex **10** through the communication
protocol stacks **22** and **38** which may function as routing
protocol stacks as described herein.

As an example of utilization of the present
invention, for illustration purposes, data processing
system **20** has associated with it communication protocol
stack **22** which is associated with MVS image MVS 1 which
has application APP A executing on MVS image MVS 1 and
utilizing communication protocol stack **22** to allow access
to, for example, client **46** through network **44**.
Similarly, data processing system **24** has associated with
it communication protocol stack **26** which is associated
with MVS image MVS 2 which has a second instance of
application APP A and an instance of application APP B
executing on MVS image MVS 2 which may utilize
communication protocol stack **26** for communications. Data
processing system **28** has associated with it communication
protocol stack **30** which is associated with MVS image MVS
3 which has a second instance of application APP B
executing on MVS image MVS 3 which may utilize
communication protocol stack **30** for communications. Data
processing system **32** has associated with it communication
protocol stack **34** which is associated with MVS image MVS
4 which has a third instance of application APP A
executing on MVS image MVS 4 which may utilize
communication protocol stack **34** for communications.
Finally, data processing system **36** has associated with it
communication protocol stack **38** which is associated with
MVS image MVS 5 which has a third instance of application

APP B executing on MVS image MVS 5 which may utilize communication protocol stack **38** for communications.

As is illustrated in **Figure 7**, a communication protocol stack may be both a routing communication protocol stack and a server communication protocol stack. Communication protocol stacks **22**, **26** and **34** may function as server communication protocol stacks for the application APP A. Likewise, the communication protocol stacks **26**, **30** and **38** may function as server communication protocol stacks for the application APP B. In the present example, communication protocol stacks **22** and **26** may be defined as a subset of the available communication protocol stacks **22**, **26** and **34** for application APP A and may be associated with particular (for example, based on source IP address) clients requesting connections to APP A.

Utilizing the above described system configuration as an example, the workload distribution function **25** utilizing the VIPA distribution function **23** will now be described. The VIPA distribution function **23** allows for protocol stacks which are defined as supporting DVIPAs to share the DVIPA and communicate with the network **44** through a routing protocol stack such that all protocol stacks having a server application which is associated with the DVIPA may appear to the network **44** as a single IP address. Such dynamically routable VIPAs may be provided by designating a protocol stack, such as protocol stack **22**, as a routing protocol stack, notifying other protocol stacks of the routing protocol stack and having other protocol stacks notify the routing protocol

stack when an application which binds to the DVIPA is started. Because communications to the DVIPA are routed through the routing protocol stack, the routing protocol stack may provide work load balancing based on policies

5   which define subsets of target communication protocol stacks through the workload distribution function **25** by distributing connections to selected ones of the protocol stacks on MVS images executing server applications which bind to the DVIPA to balance workload. The VIPA

10   distribution function **23** allows automated movement of a routing protocol function to a backup stack and allows recovery of a failed routing stack without disruption to connections through the backup stack. In such cases, the workload distribution function **25** would also be moved or

15   recovered on the backup stack.

For the present example, application APP A is associated with DVIPA VA1 which may be associated with the respective first, second and third instances of APP A. Application APP B, likewise, has DVIPA VB1 associated

20   with the respective first, second and third instances of APP B.

Returning to the example illustrated in **Figure 7**, the communication protocol stacks **22, 26, 30, 34** and **38** may be configured as to which stacks are routing stacks,

25   backup routing stacks and server stacks as described in the above referenced United States Patent Applications. The workload distribution functions **25** for routing communication protocol stacks which utilize policies which define subsets of available target communication

30   protocol stacks may perform the operations illustrated in **Figure 8.**

As seen in **Figure 8**, the workload distribution function **25** of routing communication protocol stacks, such as the communication protocol stacks **22** and **38**, periodically obtains workload information from OS/390 WLM
5   utilizing conventional Sysplex communication techniques (block **600**). This information may take the form of a workload weight (W) associated with each server communication protocol stack for which the communication protocol stacks **22** and **38** function as routing
10   communication protocol stacks. The weight may, for example, be a value between 0 and 64 with 64 representing the highest available processing capacity.

The workload distribution function **25** of the routing communication protocol stacks **22** and **38** also periodically
15   poll the other communication protocol stacks to obtain QoS information from the server communication protocol stacks (block **602**). Such a poll may be individually addressed to the server communication protocol stacks or may be broadcast. Polling may utilize XCF messaging and
20   the coupling facility **40** for communication between the communication protocol stacks. The QoS information may be provided by the policy agents **27** of the server communication protocol stacks or by the workload distribution function **25** if the stack is both a server
25   stack and a routing stack. Furthermore, if the stack is both a server stack and a routing stack, the polling of itself may not involve external communications.

The workload information and the QoS information may be obtained for only communication protocol stacks for
30   which a communication protocol stack functions as a

routing communication protocol stack or for all communication protocol stacks in the Sysplex. A communication protocol stack knows if it is a routing communication protocol stack as a result of the DVIPA

5   initialization described in commonly assigned United States Patent Application Serial No. 09/640,412, entitled, "METHODS, SYSTEMS AND COMPUTER PROGRAM PRODUCTS FOR NON-DISRUPTIVELY TRANSFERRING A VIRTUAL INTERNET PROTOCOL ADDRESS BETWEEN COMMUNICATION PROTOCOL STACKS"

10   (Attorney Docket No. 5577-207), United States Patent Application Serial No. 09/640,409, entitled "METHODS, SYSTEMS AND COMPUTER PROGRAM PRODUCTS FOR CLUSTER WORKLOAD DISTRIBUTION" (Attorney Docket No. 5577-205), United States Patent Application Serial No. 09/640,438,

15   entitled "METHODS, SYSTEMS AND COMPUTER PROGRAM PRODUCTS FOR FAILURE RECOVERY FOR ROUTED VIRTUAL INTERNET PROTOCOL ADDRESSES" (Attorney Docket No. 5577-206) all filed August 17, 2000, and United States Patent Application Serial No. 09/401,419 entitled "METHODS, SYSTEMS AND

20   COMPUTER PROGRAM PRODUCTS FOR AUTOMATED MOVEMENT OF IP ADDRESSES WITHIN A CLUSTER" filed September 22, 1999.

      For example, the communication protocol stacks **22** and **38,** which are designated as routing protocol stacks as they have connections to the network **44** and include

25   VIPADISTribute statements in the VIPADynamic block, publish the distribution information through messages broadcast by the VIPA distribute function **23** of each of the communication protocol stacks **22, 26, 30, 34** and **38** to the other communication protocol stacks **22, 26, 30,**

30   **34** and **38.** At initialization or profile changes, the

communication protocol stacks **22, 26, 30, 34** and **38** communicate to all partner communication protocol stacks the complete list of dynamic routable VIPAs, their associated *ipAddrList* and *portlist* and the primary and

5 backup definitions for the communication protocol stack. When a communication protocol stack **22, 26, 30, 34** and **38** receives the DVIPA information it notes if it is identified as a candidate target protocol stack or as a backup stack. If the protocol stack is a candidate target

10 stack, the protocol stack monitors applications associated with the protocol stack and sends a message to the defined routing stack when an application instance is bound to the DVIPA and listens on a defined port. Thus, each routing communication protocol stack is aware of

15 each communication protocol stack for which it routes DVIPAs.

Returning to **Figure 8**, when a routing communication protocol stack receives a request for a connection to an application (block **604**), the routing communication

20 protocol stack determines the characteristics of the received request (block **605**). The routing communication protocol stack determines the subset of the potential target communication protocol stacks for the connection based on the characteristics of the request (block **606**)

25 and determines a workload metric for the subset of potential target communication protocol stacks (block **608**).

Thus, in the present example, when a connection request is made to DVIPA VA1 to connect to APP A, the

30 communication protocol stack **22** evaluates the client

making the request and if the client is in a list of
clients associated with the subset of communication
protocol stacks, the communication protocol stack **22**
obtains the workload information W and QoS information
for MVS1 and MVS2. This information is used to determine
the workload metrics for the candidate target protocol
stacks in the subset of potential target communication
protocol stacks. The workload metric may, optionally, be
evaluated to determine if a stack in the subset of stacks
meets a performance or availability criteria (*e.g.*, is
the workload metric above a threshold value)(block **609**).
If so, the workload metric may, for example, combine the
QoS information and the workload information to select
the stack in the subset of stacks with the best relative
performance (*e.g.*, highest weight)(block **610**).

If no stack in the subset of stacks meets the
performance or availability criteria (block **609**), then
the workload metric is determined for the other potential
target communication protocol stacks (block **614**). In the
present example, the workload metric would be determined
for MVS4. The workload metrics for all of the potential
target communication protocol stacks could then be
evaluated to select the stack with best workload metric
(*e.g.* highest value) from all of the communication
protocol stacks (block **616**). Alternatively, the workload
metrics for the stacks outside the subset could be
compared to the availability criteria and only selected
if they met the availability criteria. Otherwise a stack
with the best workload metric from the subset of stacks

would be selected. In either case, the requested connection is routed to the selected stack (block **612**).

Alternatively, if no stack in the subset of stacks meets the performance or availability criteria (block **609**), the request could be rejected. In such embodiments, the operations of blocks **614** and **616** could be eliminated and the "NO" path from block **609** could be connected to block **612**. With no server stack selected, then the connection would not be routed to any stack and would be rejected (block **612**).

The combination of the QoS information and the workload information may be as described in concurrently filed and commonly assigned United States Patent Application Serial No. _____, entitled "METHODS, SYSTEMS AND COMPUTER PROGRAM PRODUCTS FOR WORKLOAD DISTRIBUTION BASED ON END-TO-END QUALITY OF SERVICE" (Attorney Docket No. 5577-214), the disclosure of which is incorporated herein by reference as if set forth fully herein.

Embodiments of the present invention have been described with reference to **Figures 4** through **8** which are block diagrams and flowchart illustrations of operations of protocol stacks incorporating embodiments of the present invention. It will be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, can be implemented by computer program instructions. These program instructions may be provided to a processor to produce a machine, such that the instructions which execute on the processor create

means for implementing the functions specified in the flowchart and/or block diagram block or blocks. The computer program instructions may be executed by a processor to cause a series of operational steps to be performed by the processor to produce a computer implemented process such that the instructions which execute on the processor provide steps for implementing the functions specified in the flowchart and/or block diagram block or blocks.

Accordingly, blocks of the flowchart illustrations and/or block diagrams support combinations of means for performing the specified functions, combinations of steps for performing the specified functions and program instructions for performing the specified functions. It will also be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, can be implemented by special purpose hardware-based systems which perform the specified functions or steps, or combinations of special purpose hardware and computer instructions.

While the present invention has been described with respect to the workload distribution function as a part of the communication protocol stack, as will be appreciated by those of skill in the art, such functions may be provided as separate functions, objects or applications which may cooperate with the communication protocol stacks. Furthermore, the present invention has been described with reference to particular sequences of operations. However, as will be appreciated by those of skill in the art, other sequences may be utilized while

still benefitting from the teachings of the present invention. Thus, while the present invention is described with respect to a particular division of functions or sequences of events, such divisions or

5 sequences are merely illustrative of particular embodiments of the present invention and the present invention should not be construed as limited to such embodiments.

Furthermore, while the present invention has been

10 described with reference to particular embodiments of the present invention in a System/390 environment, as will be appreciated by those of skill in the art, the present invention may be embodied in other environments and should not be construed as limited to System/390 but may

15 be incorporated into other systems, such as a Unix or other environment, for example, by associating applications or groups of applications with an address rather than a communications adapter. Thus, the present invention may be suitable for use in any collection of

20 data processing systems which allow sufficient communication to all systems for the use of dynamic virtual addressing. Accordingly, specific references to System/390 systems or facilities, such as the "coupling facility," "ESCON," "Sysplex" or the like should not be

25 construed as limiting the present invention.

In the drawings and specification, there have been disclosed typical preferred embodiments of the invention and, although specific terms are employed, they are used in a generic and descriptive sense only and not for

30 purposes of limitation, the scope of the invention being set forth in the following claims.